Commercial CPU and SOAC Development

Rajan Bedi

Senior Design Engineer

Advanced RISC Machines (ARM) Ltd.

Agenda

- ARM (IP) Business Model
- RISC CPU Architecture
- ARM CPU Roadmap - ARM7, 8 & 9 families
- System On A Chip
- AMBA on-chip system bus
- ARM CPU Roadmap - ARM10 and 11 families
- CPU and SOAC Verification
- Future Research
- Questions

ARM Business Model

- *ARM is a fabless chip company that licenses its IP to most of the major semiconductor companies.*
- *ARM designs low-power, high-performance 32-bit RISC CPUs and System-on-chip solutions based around its CPU architecture.*
- *ARM designs 'Hard' and 'Soft' IP.*
  - *Hard IP - fully characterized on a target process and exploits the area, power and performance advantages of full-custom, hand-tuned layout.*
  - *Soft IP - delivery of a synthesizable netlist from RTL.*
- *Our IP is used primarily in embedded products.*

ARM Technology for Everyone

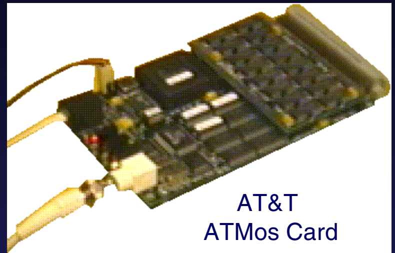
Wireless

TOSHIBA
SANYO
NEC
HITACHI
Panasonic
SAMSUNG
SONY
MOTOROLA
KENWOOD
KYOCERA
NOKIA
CONNECTING PEOPLE
ERICSSON
QUALCOMM
DENSO
ALCATEL

Telecom/Networking

ERICSSON

Cayman Systems' ADSL3220 Ethernet to Internet Router



AT&T ATMos Card

DECT Gigaset

SIEMENS

G.lite Router

Mapletree Networks' 1000 Series Remote Access Concentrator

Parrot WLAN

netopia

R7200 SDSL Router

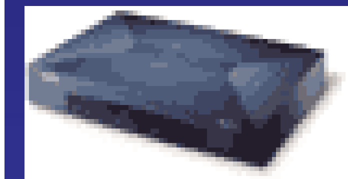
Alcatel's SPEED TOUCH Home ADSL USB Modem

NeoNetworks' StreamProcessor 2400 Backbone Switch Router

NETsilicon's 10/100 Ethernet Pocket Print Server

3Com's 10/100 PCI NIC

Rebel.com Netwinder



ZyXEL's Prestige 641 ADSL Router

Ethernet ADSL Router

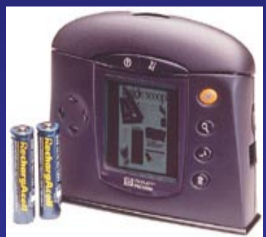
Imaging

TOSHIBA

brother

SAMSUNG

WizCom



HEWLETT PACKARD

Consumer Electronics

CREATIVE NOMAD WORLD

COMPAQ

SEGA

PACE

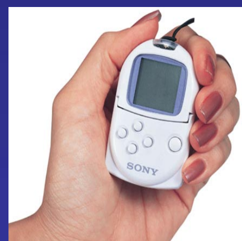
PSION

met@box

NOKIA

ASKEY

Palm PC

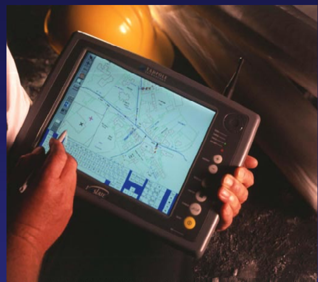


SONY

Automotive & Navigation

empeg

In-Car Digital Player



TADPOLE TECHNOLOGY

Cartesian J -Slate

SiRF

GPS Module

SUMITOMO ELECTRIC

MARCONI CANADA

CMT-1200 ALLSTAR GPS

Raytheon

Marine
Pathfinder Radar

DENSO

The ARM Partnership

RISC CPU Architecture Characteristics

- *Load/Store architecture*
- *Large Register Bank - typically thirty two 32-bit registers*
- *Fixed size for all instructions - 32 bits long*
- *Pipelined execution*
- *Single cycle execution*
- *Orthogonal Instruction Set*
- *Hardwired instruction decode logic*

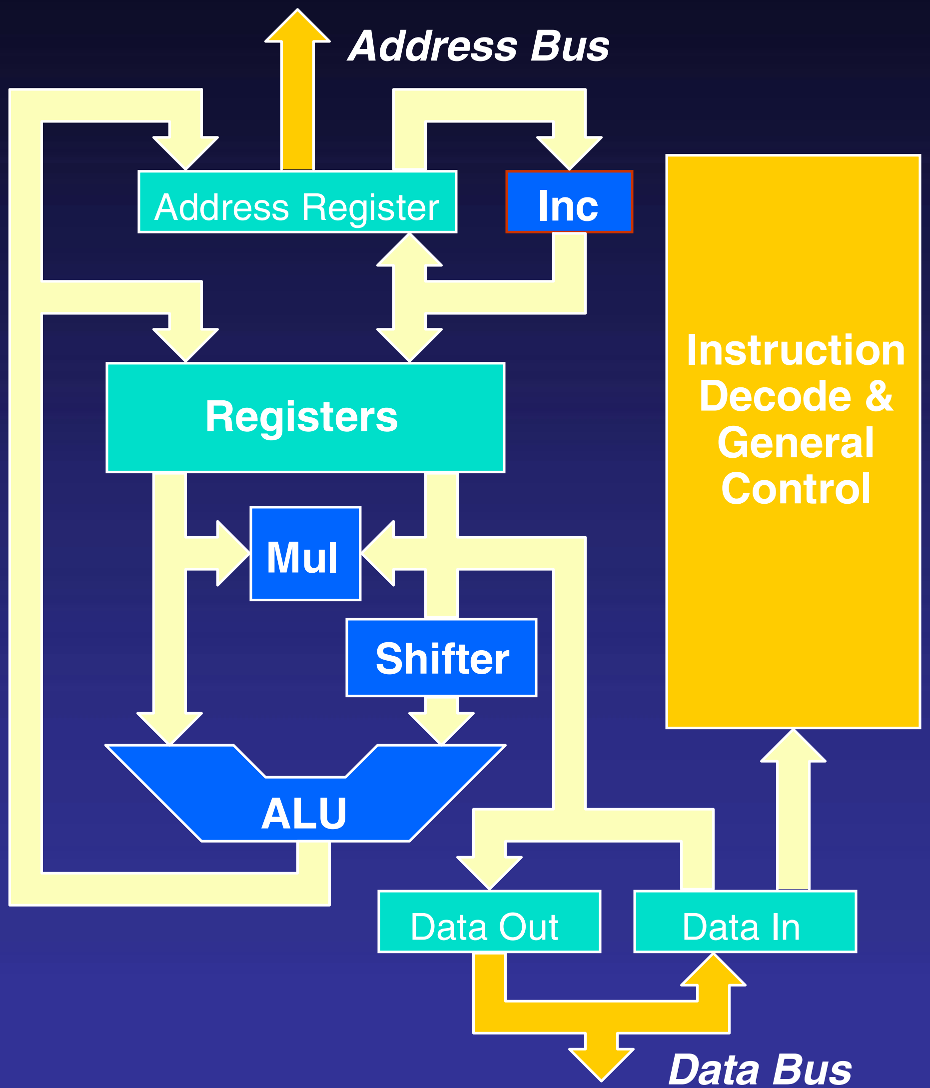
ARM7 32-bit RISC Architecture

- *Von Neumann Enhanced RISC Architecture*
- *Three Stage Pipeline - Fetch, Decode & Execute*
- *Conditional execution of every instruction*
- *32-bit flat address space (4 GB memory map)*
- *Most instructions execute in a single cycle.*
- *Combined ALU and shifter for high speed bit manipulation*

ARM7 RISC Architecture (cont.)

- *Powerful multiple load and store instructions combined with auto-indexing addressing modes*
  - *Block Copy*
  - *Stack Manipulation*
- *Open instruction set extension via coprocessors*

ARM7 block diagram



In a single-cycle data processing instruction, two register operands are accessed, the value on the B bus can be shifted and is combined with the value on the A bus in the ALU. Result is written back into the register bank. PC value is in the address register, from where it is fed into the incrementer and then copied back into R15 and the address register to be used as the address for the next fetch.

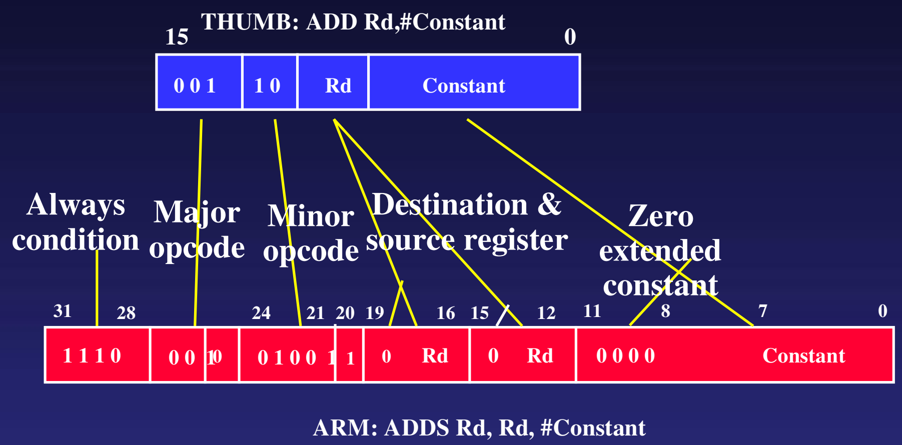
ARM register organisation

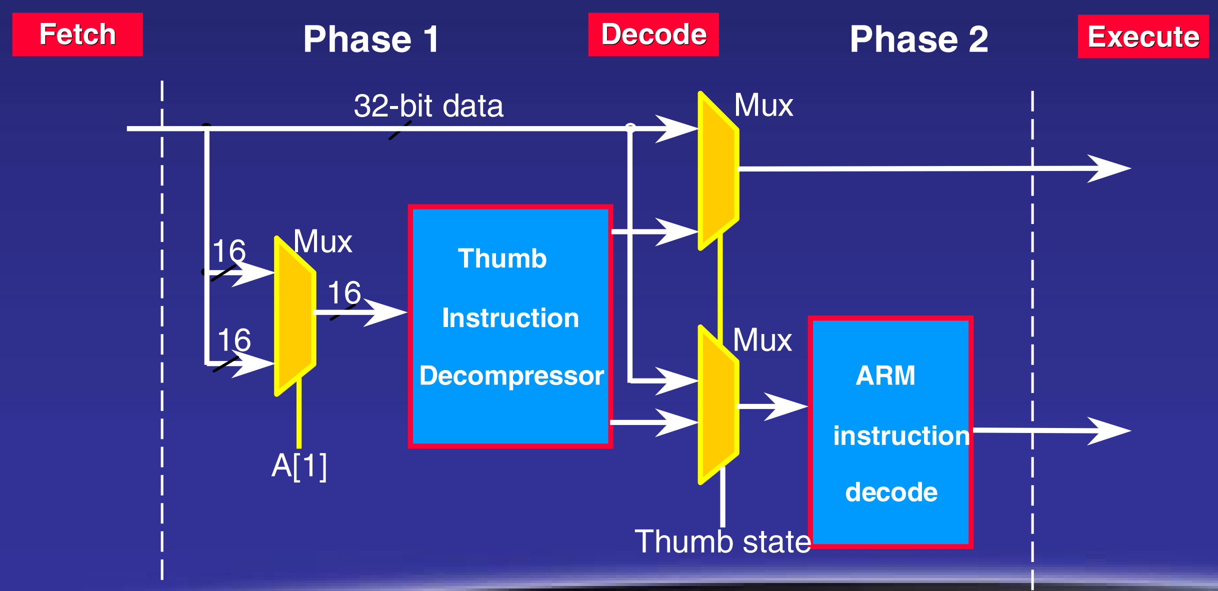
	User Mode	FIQ Mode	IRQ Mode	Supervisor Mode	Abort Mode	Undef Mode
	General Purpose Registers					
	R0	R0	R0	R0	R0	R0
	R1	R1	R1	R1	R1	R1
	R2	R2	R2	R2	R2	R2
	R3	R3	R3	R3	R3	R3
	R4	R4	R4	R4	R4	R4
	R5	R5	R5	R5	R5	R5
	R6	R6	R6	R6	R6	R6
	R7	R7	R7	R7	R7	R7
	R8	R8_FIQ	R8	R8	R8	R8
	R9	R9_FIQ	R9	R9	R9	R9
	R10	R10_FIQ	R10	R10	R10	R10
	R11	R11_FIQ	R11	R11	R11	R11
	R12	R12_FIQ	R12	R12	R12	R12
SP	R13	R13_FIQ	R13_IRQ	R13_SVC	R13_ABORT	R13_UNDEF
LR	R14	R14_FIQ	R14_IRQ	R14_SVC	R14_ABORT	R14_UNDEF
	Program Counter					
R15	PC	PC	PC	PC	PC	PC
	Program Status Registers					
	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
		SPSR_FIQ	SPSR_IRQ	SPSR_SVC	SPSR_ABORT	SPSR_UNDEF

Operating Modes / Exception Model

- The ARM has seven operating modes
  - User (unprivileged mode that most tasks run in)
  - System (privileged mode that privileged tasks run in)
  - FIQ (entered when a high priority (fast) interrupt is raised)
  - IRQ (entered when a low priority (normal) interrupt is raised)
  - Supervisor (entered when a Software Interrupt instruction is executed)
  - Abort (used to handle memory access violations)
  - Undef (used to handle undefined instructions)
- User and System mode share one bank of registers
- FIQ mode has a private R8 to R14(LR)
- All the other modes have a private R13(SP) and R14(LR)

Thumb Code Compression





Thumb Code Example

In C:

```
int iabs(int x)
        {
         if (x>=0) return x;
           else return -x;
        }
```

In ARM Code

```
CMP      r0,#0
RSBLT    r0,r0,#0
MOV      PC,lr
```

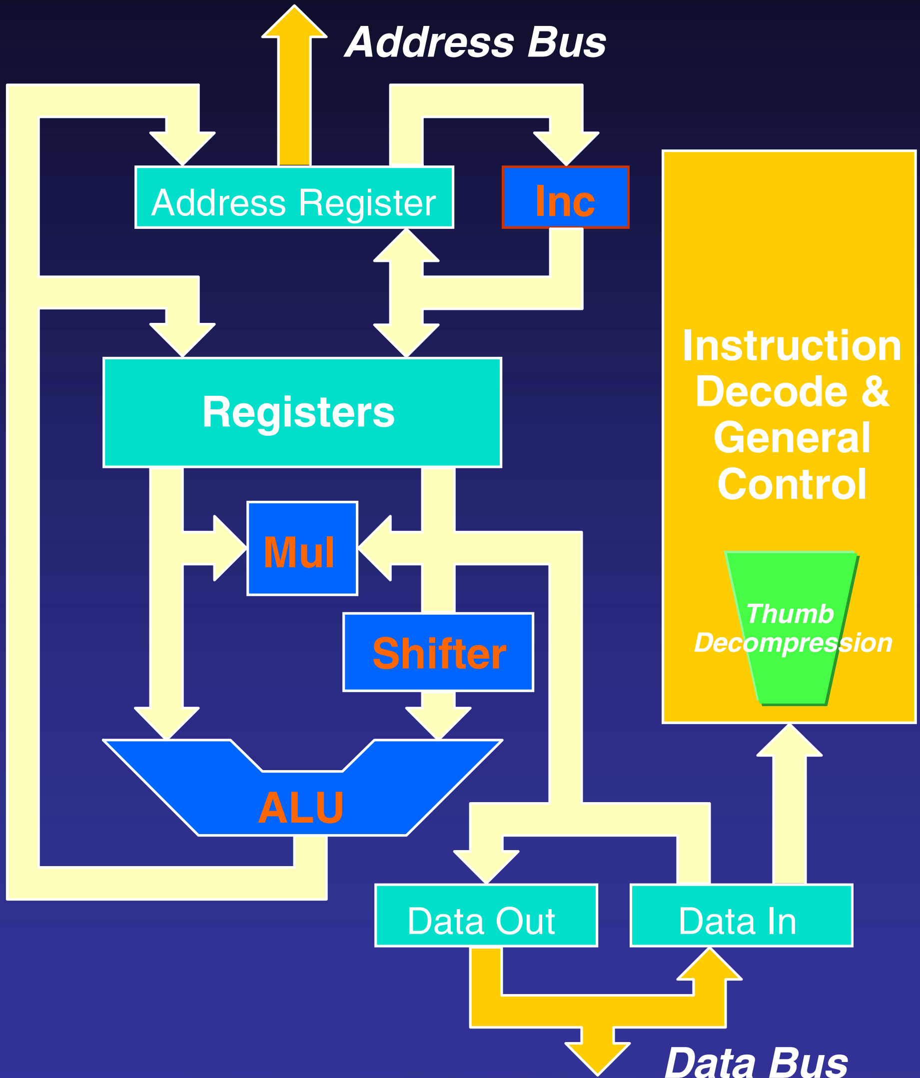
(12 bytes)

In Thumb code

```
CMP      r0,#0
BGE      return
NEG      r0,r0
return
MOV      PC,lr
```

(8 bytes 67%)

ARM7TM block diagram



Thumb Features

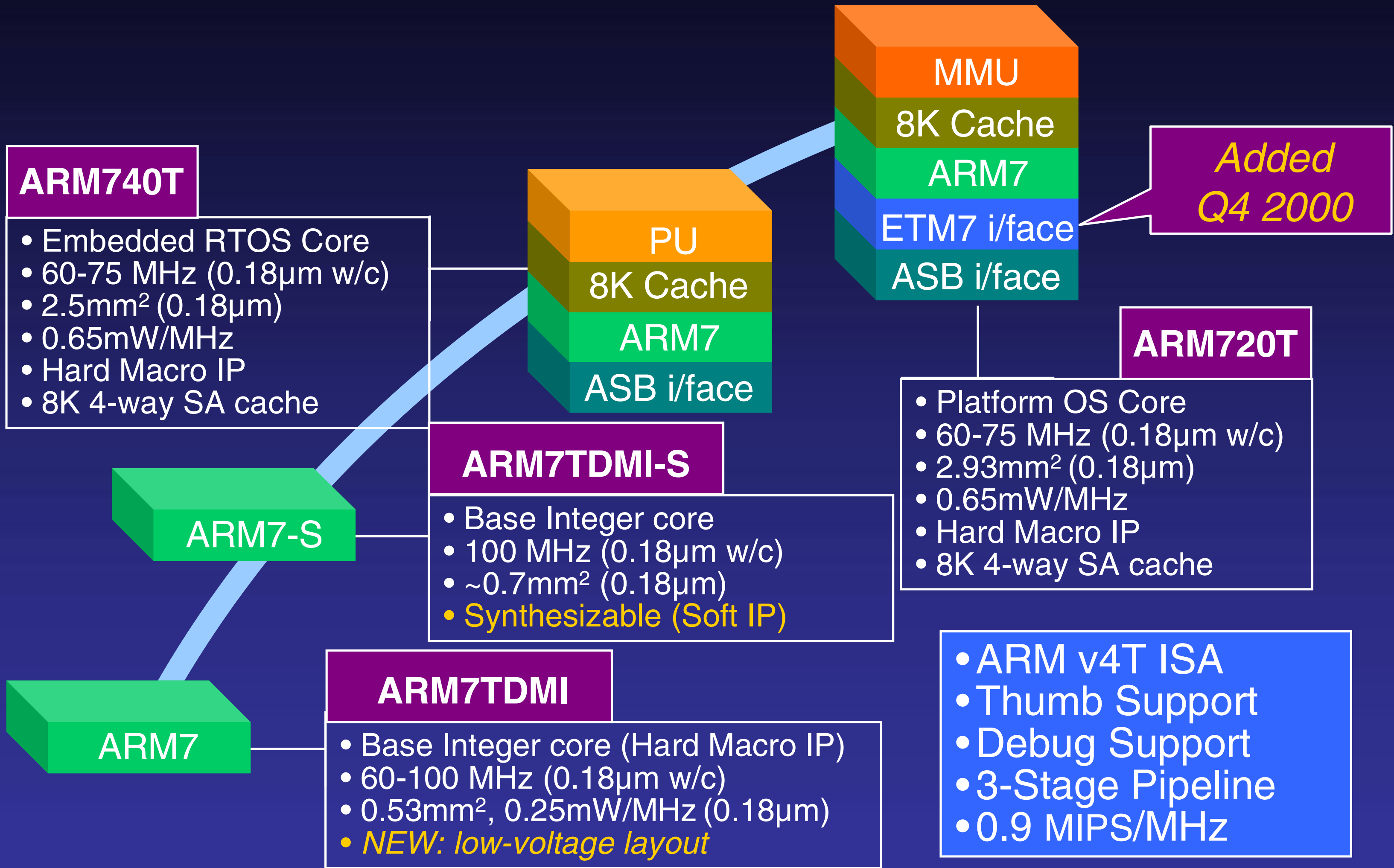
- *Thumb addresses code density*
- *All Thumb instructions are 16 bits long*
- *Thumb may be viewed as a compressed form of a subset of the 32-bit ARM instruction set.*
- *Implementations of Thumb use dynamic compression in an ARM instruction pipeline. This logic translates the 16-bit Thumb instruction into its equivalent 32-bit ARM instruction.*
- *Decompression logic added without compromising cycle time or pipeline latency - Original ARM7 pipeline did very little work in phase one of the decode cycle.*
- *Programmer's Model - r0-r7, r13, r15*

Thumb Applications

A typical embedded system, e.g. a mobile phone, will include a small amount of fast 32-bit memory (to store speed-critical DSP code) and 16-bit off-chip memory to store the control code.

- *Thumb code requires 70% of the space of the ARM code*
- *Thumb code uses 40% more instructions than ARM code*
- *With 32-bit memory, the ARM code is 40% faster than Thumb code*
- *With 16-bit memory, the Thumb code is 45% faster than ARM code*
- *Thumb code uses 30% less external memory power than ARM code*

ARM7 Family: 60-100MIPS



- ARM v4T ISA
- Thumb Support
- Debug Support
- 3-Stage Pipeline
- 0.9 MIPS/MHz

Stated operating frequencies represent worst-case speed for a range of processes.

ARM 8 Architecture

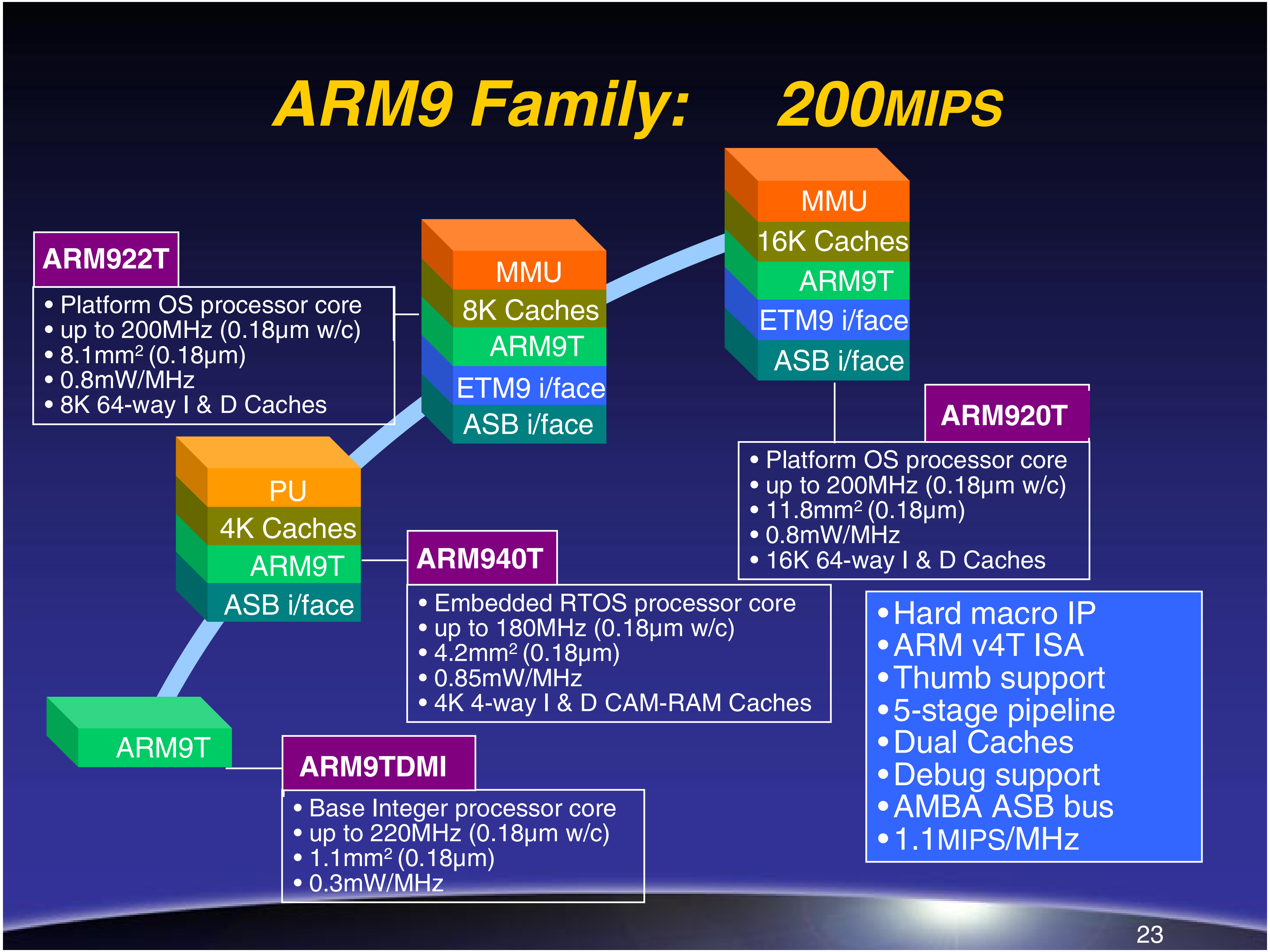
Performance of the ARM7 Von Neumann architecture limited by the available bandwidth - memory accessed on almost every cycle either to fetch an instruction or to transfer data.

- *Higher performance core than ARM7*
- *$T_{prog} = (N_{inst} * CPI) / f_{clk}$*
- *Two ways to increase performance*
  - *Increase clock rate, f_{clk} -> no. of pipeline stages increased to 5*
  - *Reduce CPI -> exploits sequential nature of memory accesses*
- *Von Neumann unified memory*
- *Double bandwidth from single memory access*
- *Prefetch buffer incorporating static branch prediction*
- *ARM8 architecture superseded by ARM9*

ARM9 Architecture

- Higher performance core than ARM8
- $T_{prog} = (N_{inst} * CPI) / f_{clk}$
- Two ways to increase performance
  - Increase clock rate, f_{clk}
  - Reduce CPI

- Five-stage pipeline - Fetch, Decode, Execute, Memory, and Write
- Interlocked pipeline with data forwarding
- Harvard architecture improves CPI


ARM9 Family: 200MIPS
ARM922T
• Platform OS processor core
• up to 200MHz (0.18µm w/c)
• 8.1mm² (0.18µm)
• 0.8mW/MHz
• 8K 64-way I & D Caches
MMU
8K Caches
ARM9T
ETM9 i/face
ASB i/face
MMU
16K Caches
ARM9T
ETM9 i/face
ASB i/face
ARM920T
• Platform OS processor core
• up to 200MHz (0.18µm w/c)
• 11.8mm² (0.18µm)
• 0.8mW/MHz
• 16K 64-way I & D Caches
PU
4K Caches
ARM9T
ASB i/face
ARM940T
• Embedded RTOS processor core
• up to 180MHz (0.18µm w/c)
• 4.2mm² (0.18µm)
• 0.85mW/MHz
• 4K 4-way I & D CAM-RAM Caches
ARM9T
ARM9TDMI
• Base Integer processor core
• up to 220MHz (0.18µm w/c)
• 1.1mm² (0.18µm)
• 0.3mW/MHz
• Hard macro IP
• ARM v4T ISA
• Thumb support
• 5-stage pipeline
• Dual Caches
• Debug support
• AMBA ASB bus
• 1.1MIPS/MHz
23

ARM9E Architecture

- DSP extensions added to ARM9 architecture
  - single cycle 16x16 and 32x16 fixed point multiply & multiply-accumulate instructions
  - saturated and signed addition and subtraction
  - count leading zeros instruction added
- Enhanced multiplier for DSP performance
- Supports 32-bit ARM, 16-bit Thumb and DSP
- Non-stop debug hardware debug - allows critical exception handlers to execute during debugging

Benefits of ARM9E

- *Single engine for both DSP and control code*
- *Simple single memory system*
- *Reduced chip complexity, die size and power consumption*
- *Single toolkit support with ARM's Development and Debug tools, giving faster time-to-market*
- *Ease of software design for MCU & DSP convergence applications - no partitioning*
- *Synthesizable IP delivery*

ARM9E Market

- *The ARM9E addresses many applications requiring a mix of DSP and control performance*
  - *Mass storage & Servo control in HDD & DVD*
  - *Speech coders - supporting multiple standards*
  - *Networking applications - including G.723.1 for voice over IP*
  - *Automotive control applications*
  - *Modems*
  - *Audio decoding (Dolby Digital, MP3, etc….)*

DSP Benefits

- Cellular Codecs
  - *Off-loading the voice processing to ARM makes a more balanced system*
  - *Efficient implementation of digital cellular speech codec*

- Personal Audio
  - *MP3 decoding takes just 11% of an ARM9E-S at 160MHz*
  - *Dolby Digital (AC3) takes just 22% of ARM9E-S at 160MHz*

- Voice Over IP
  - *G.723.1 full-duplex takes 25% of ARM9E-S at 160MHz*

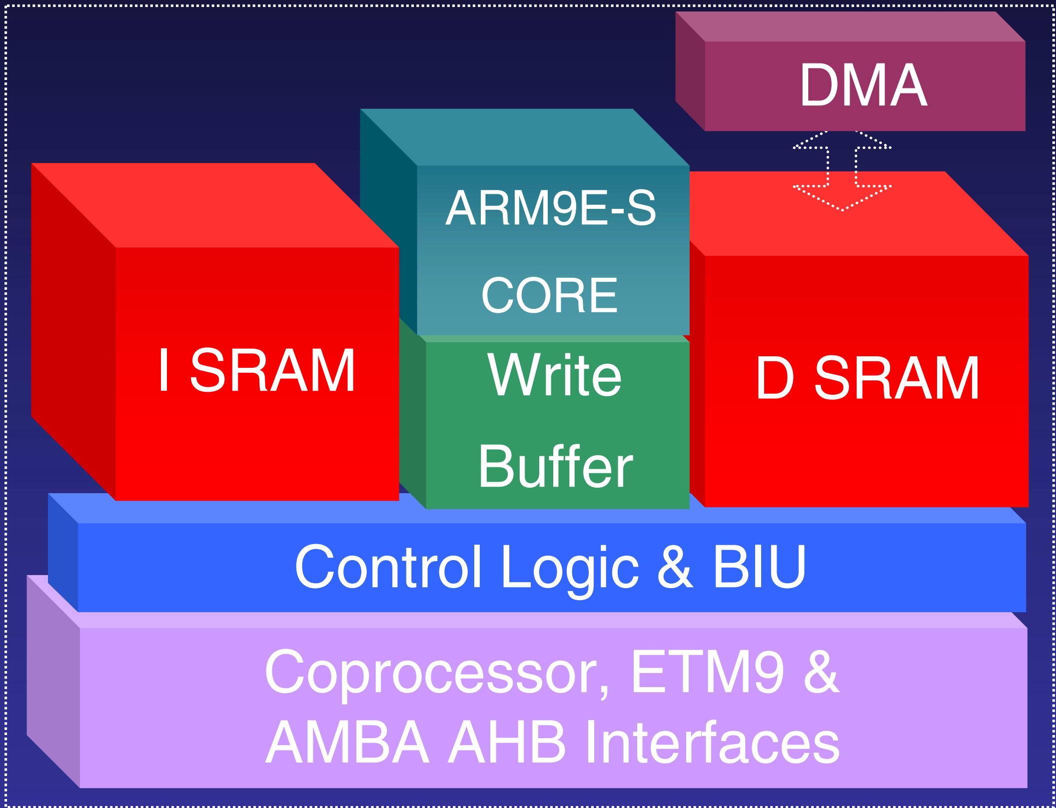
DSP Benefits

- *Single engine internet appliance*
  - *Browser, TCP/IP, Modem, Voice codec*

- *V.34bis softmodem*
  - *28% of ARM9E-S at 160MHz*

- ***Servo Controls***
  - *HDD/DVD read-write head servo control*
  - *Industrial motors*

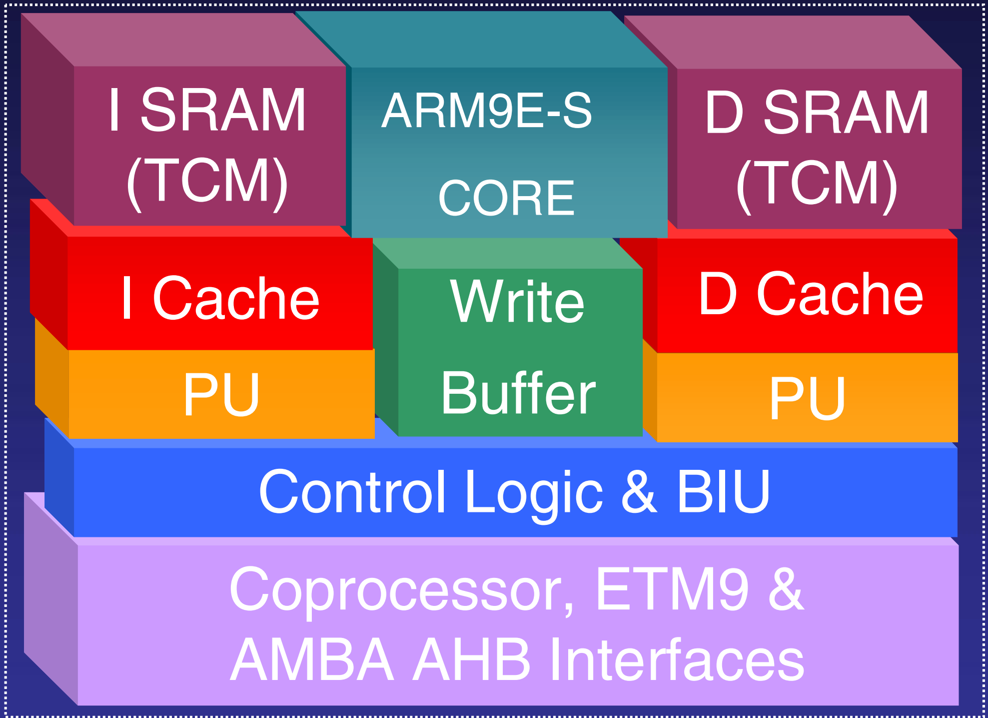
ARM966E-S



Solution for hard real time applications

- *ARM9E-S core (v5TE ISA)*
- *Write buffer*
- *Selectable size Instruction and data RAM (0kB - 1MB) for deterministic performance*
- *DMA port direct into D SRAM*
- *AMBA AHB bus interface*
- *Provides an "off-the-shelf" standard ARM9E-S solution*
- *Can attach Embedded Trace Module (ETM9) for real time trace*

ARM946E-S



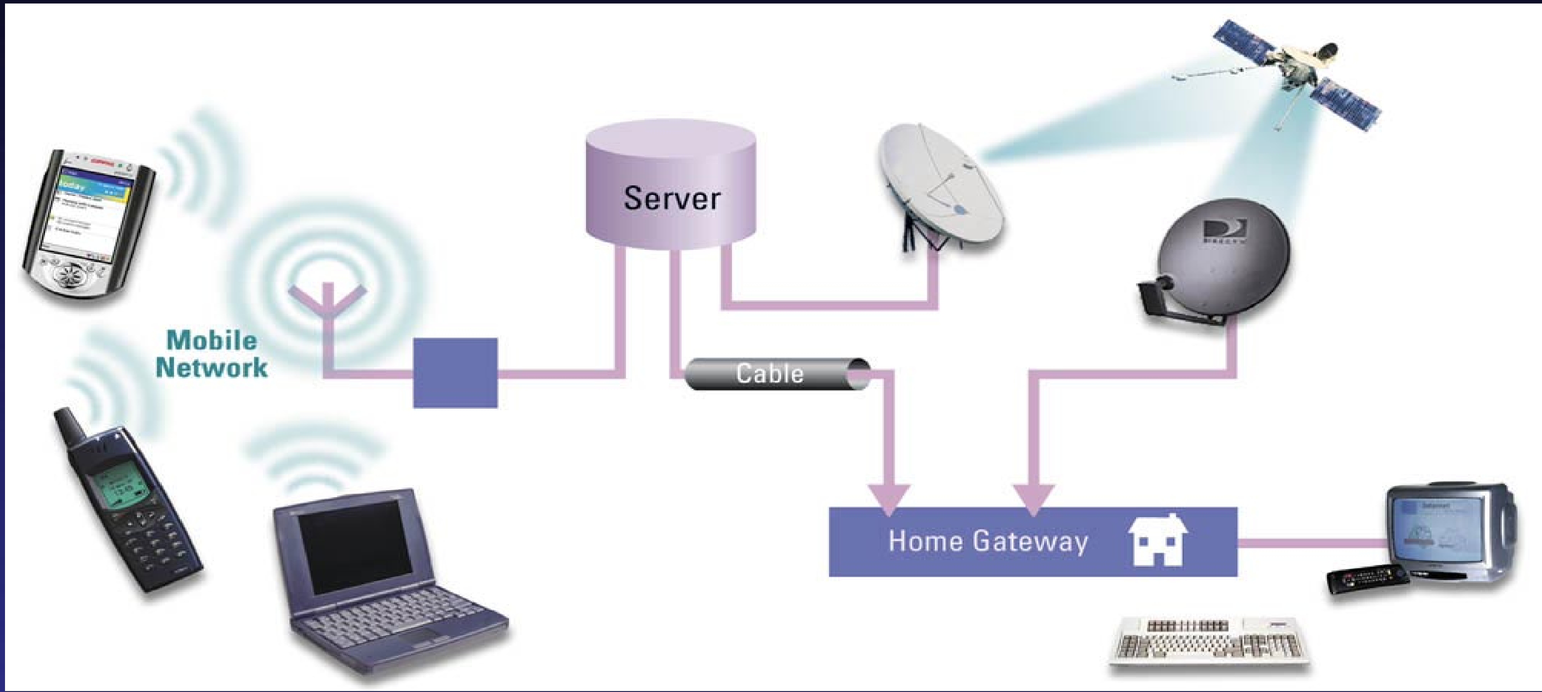
Cached Embedded Processor Solution

- *ARM9E-S core (v5TE ISA)*
- *Write buffer*
- *Memory Protection Unit*
- *Selectable size Instruction and data caches (0kB - 1MB)*
- *Selectable size Instruction and data RAM (0kB - 1MB)*
- *Designed for RTOS*
- *AMBA AHB bus interface*
- *Can attach Embedded Trace Module (ETM9) for real time trace*

ARM946E-S Cache

- *Cache is 4-way set associative*
  - *Can be built with compiled ASIC RAM*
- *Sizes of 0K, 4KB, 8KB… 1MB supported*
  - *I and D Cache sizes independently selectable*
- *Cache lock-down on per-set basis*
  - *So granularity is 1/4 of cache size*
- *Software selectable replacement algorithm*
  - *Pseudo-random and round-robin*
- *Write through and write back s/w selectable*
- *Line length fixed at 8 words*

Java - Enabling New Markets



Jazelle is an extension to the ARM architecture that allows the direct execution of JVM bytecodes. Stack-based operations are executed directly by the core and more complex bytecodes, e.g. method invocation, are executed in software.

In Java mode, an extra decode pipeline decompresses the Java bytecode stream into ARM instructions that are then issued into the ARM decode stage.

Java - Enabling New Markets

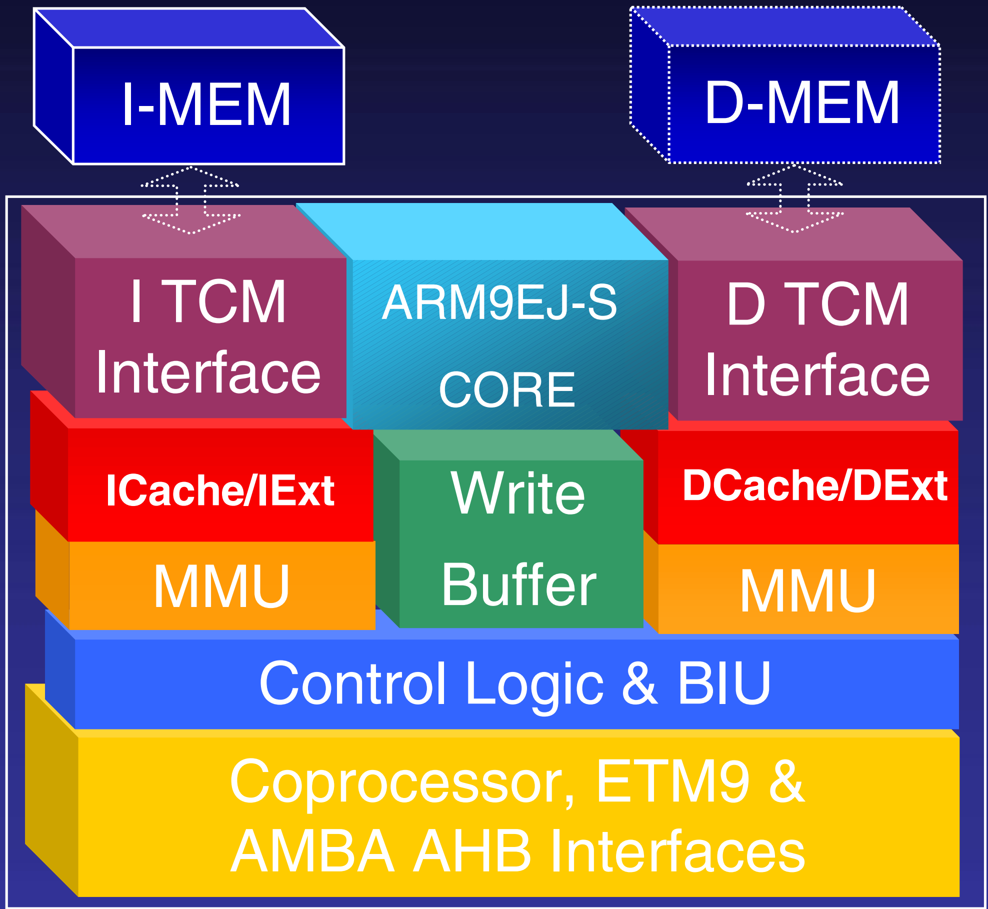
- *Ability to down-load and run portable applications over internet enables new products & services for consumer & corporate markets :*
  - *web-pages and program guides with interactive content,*
  - *shopping services, banking/financial services,*
  - *office applications, and enterprise computing applications….*
  - *games*

- *Java enables new markets for*
  - *telecom service providers*
  - *cable & satellite TV service providers*
  - *retailers*
  - *financial services providers*

Jazelle^{TM} - JAVA Extension to ARM Architecture

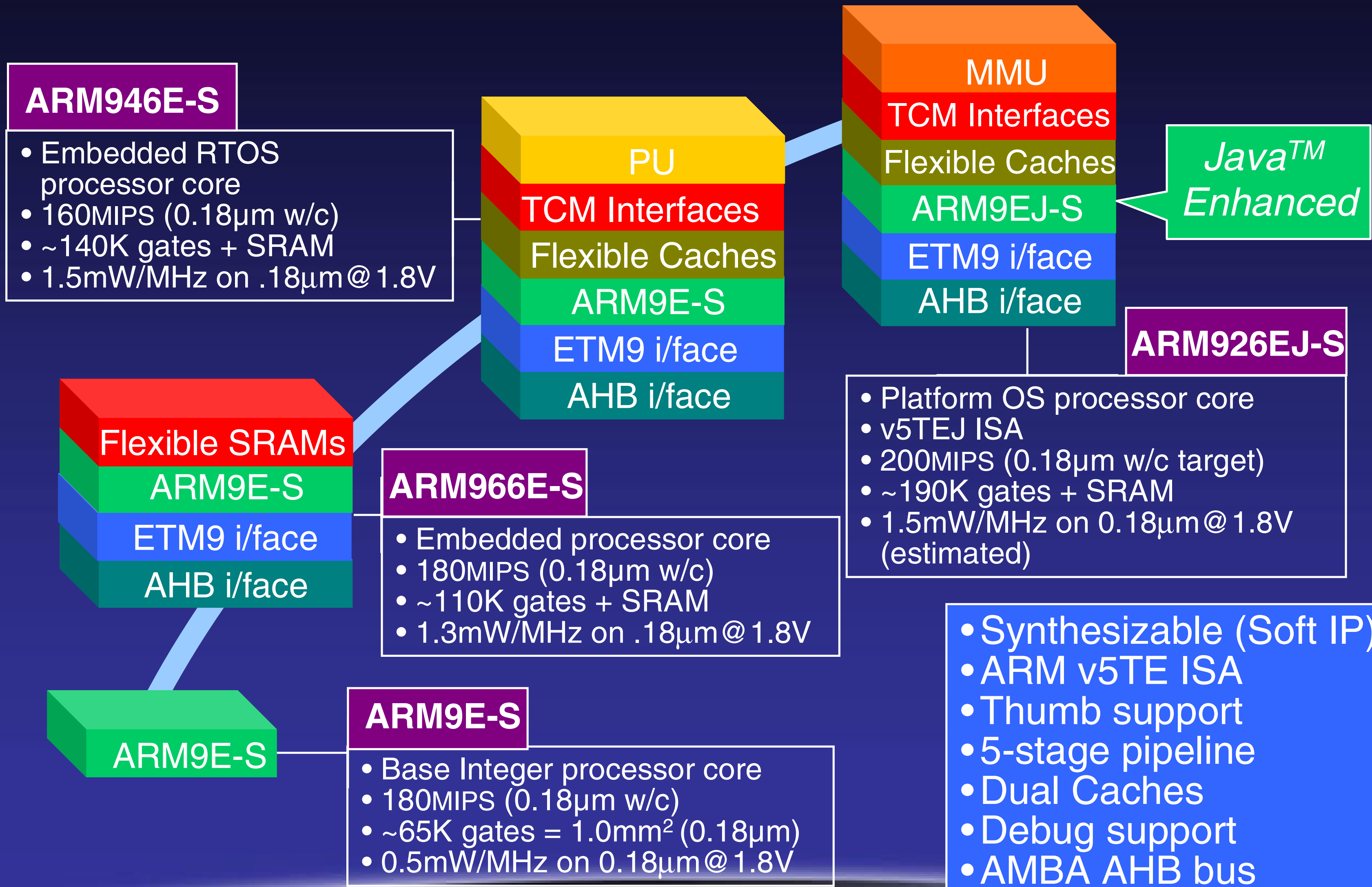
- Direct execution of JVM ByteCodes
- An ARM Instruction Set Architecture (ISA) with
  - One new instruction to enter Java state
  - Hardware emulation of Java Virtual Machine
  - Direct execution of over 80% of executed byte codes
  - Traps “complex” Byte Codes to Support Code
- 8x Performance of Software JVM
  (Embedded CaffeineMark3.0)
- Single Processor for Java and existing OS and applications
- Supported by leading Java Run-time environments
- 32-bit ARM, 16-bit Thumb and 8-bit JVM bytecodes

ARM926EJ-S for Platform OS and Java execution



- *Jazelle™ Enhanced Cached Processor for Platform OS Applications*
  - *MMU to support: EPOC, PalmOS, WindowsCE & Linux*
  - *Selectable size instruction & data cache (0 - 1MB)*
  - *Instruction & data Tightly Coupled Memory (TCM) interfaces*
  - *Target speed - up to 200MHz (w/c) on TSMC standard 0.18µm*
  - *DSP Instruction set*

ARM9E Family: 200MIPS + DSP



- Synthesizable (Soft IP)
- ARM v5TE ISA
- Thumb support
- 5-stage pipeline
- Dual Caches
- Debug support
- AMBA AHB bus

AMBA - Advanced Microcontroller Bus Architecture

Open architecture to standardize the on-chip connection of different macrocells.

- *To facilitate the right-first-time development of embedded microcontrollers with one or more CPUs.*
- *To be technology-independent and ensure that highly reuseable peripheral and system macrocells can be migrated across a diverse range of IC processes.*
- *To encourage modular system design to improve processor independence, allowing for the development of advanced cached CPU cores and peripheral libraries.*
- *To minimise the silicon infrastructure required to support efficient on-chip and off-chip communication for both operation and manufacturing test.*

AMBA - Advanced Microcontroller Bus Architecture

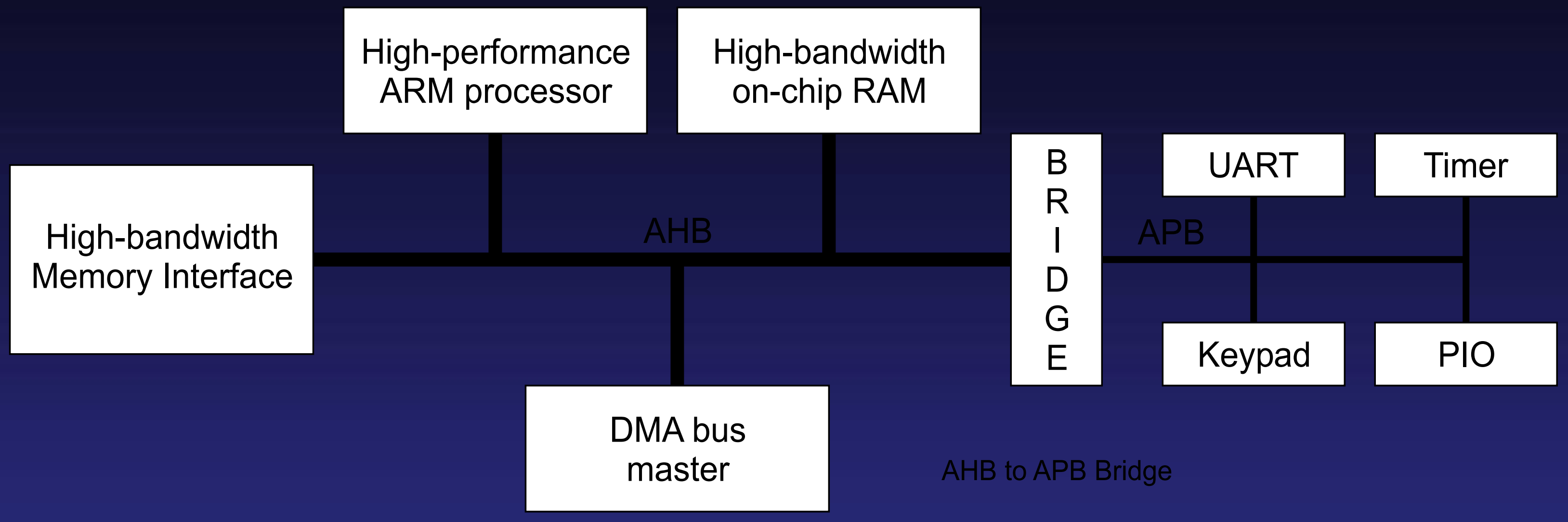
Three distinct buses defined by the AMBA Specification

- *AMBA AHB - High performance, high frequency system modules*
  - *Pipelined Operation, Multiple Bus Masters*
  - *Burst Transfers, Split Transactions*
  - *Suitable for synthesis flows and automatic test techniques*

- *AMBA ASB - High performance system modules*
  - *Pipelined Operation, Multiple Bus Masters*

- *AMBA APB - Low bandwidth peripherals*
  - *Low Power, Latched address and control, Simple Interface*
  - *Suitable for many peripherals*

Typical AMBA System



AMBA Advanced High-performance Bus (AHB)
* High performance
* Pipelined operation
* Burst transfers
* Multiple bus masters
* Split transactions

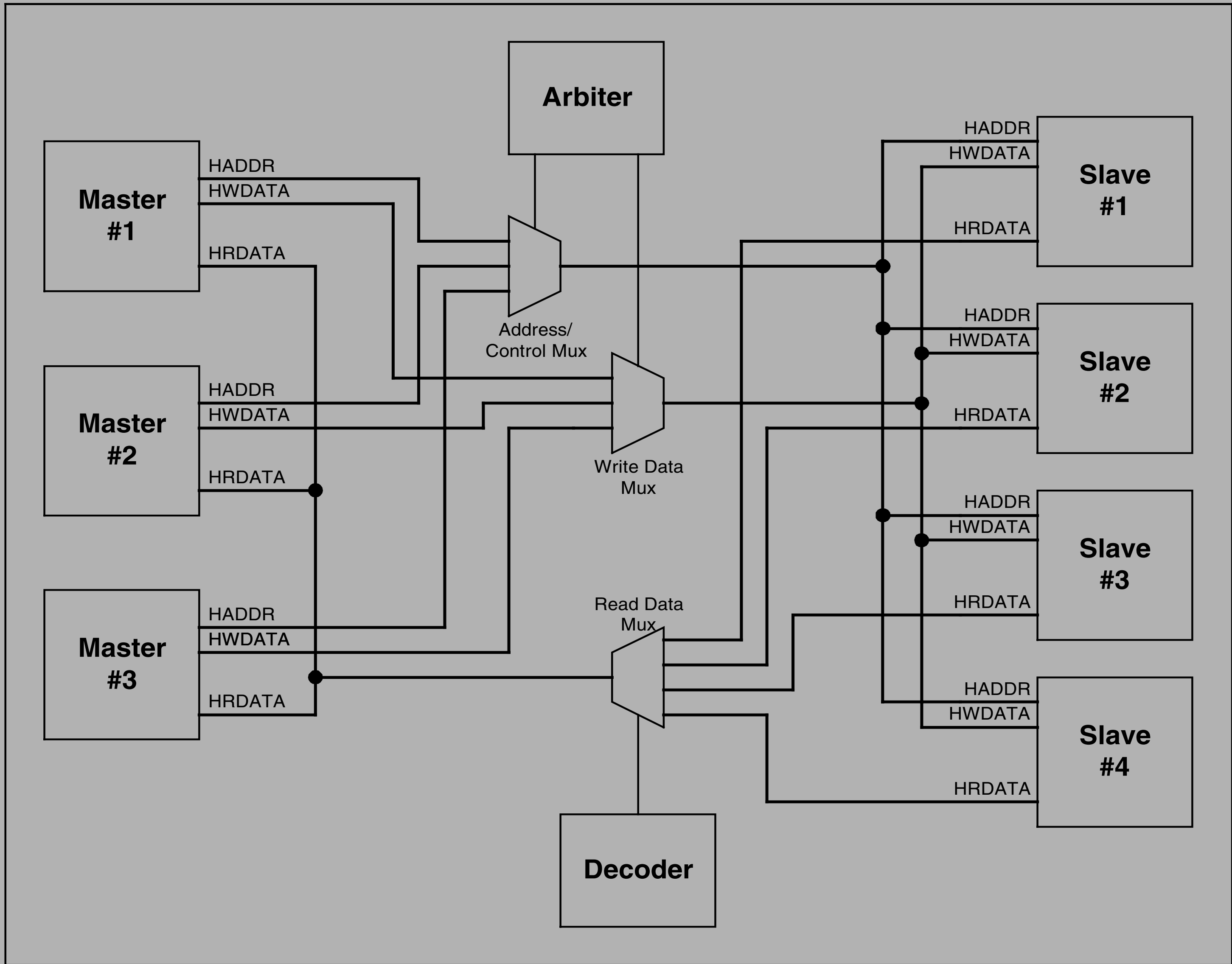
AMBA Advanced Peripheral Bus (APB)
* Low power
* Latched address and control
* Simple interface
* Suitable for many peripherals

An AMBA-based microcontroller consists of a high-performance backbone bus (AHB or ASB), able to sustain the external memory bandwidth which the CPU, on-chip memory and other DMA devices require.

Structure of AHB Design

- *AHB bus protocol has been designed to be used with a central MUX interconnection scheme.*
- *All bus masters drive out the address and control signals indicating the transfer they wich to perform.*
- *Arbiter determines which bus master has the highest priority and routes its address and control signals to all the slaves.*
- *Central decoder controls the read data and slave responses and selects the appropriate signals from the slave involved in the transfer.*

Structure of AHB Design



Structure of AHB Design

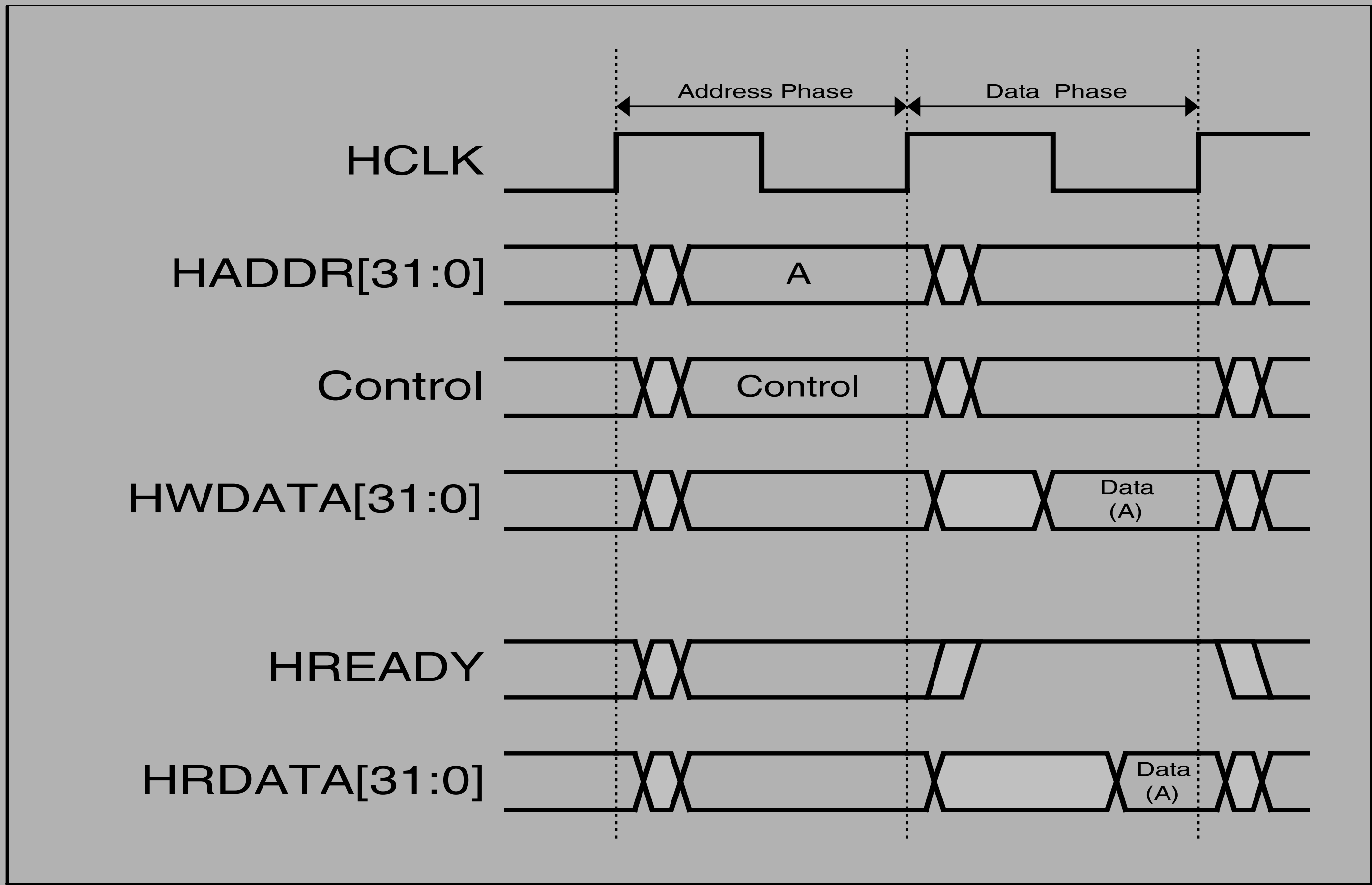
- *Before an AHB transaction can commence, the bus master must be granted access to the bus. This process is started by the master asserting a request signal to the arbiter.*

- *A granted bus master starts an AHB transaction by driving the address and control signals.*

- *AHB Arbiter ensures only one bus master has access to the bus at any one time.*

- *AHB Decoder provides a select signal for each slace attached to the bus. Centralized decoding improves the portability of peripherals by making them independent of the memory map.*

Basic Pipelined AMBA AHB Transfer



Bus master drives address and control signals onto bus at rising edge of HCLK

Slave macrocell samples address and control signals at the next rising edge

Slave drives appropriate response, sampled by master on the next rising edge

PrimeCell Peripherals

A range of reusable 'plug-and-play' AMBA compliant and VSIA compliant peripherals

PrimeCell Peripherals List

Part # ARM	Description	AMBA bus Type	Availability	Gate Count
PL011	UART w/ IrDA SIR: Similar to16C550, 16Byte FIFO, Modem and flow control, up to 115K2 bits/s	APB	Now	7.4 k
PL021	Synchronous Serial I/F (Master & Slave) : Supports Motorola SPI, TI SSI, Microwire	APB	Now	8.0 k
PL030	Real Time Clock : 32 Bit Counter, Match Reg, Requires 1Hz Ck	APB	Now	2.8 k
PL050	Keyboard/mouse I/F: PS/2 compatible	APB	Now	2.0 k
PL060	General Purpose IO: 2x8bit	APB	Now	0.8 k
PL160	DC to DC Converter Interface: 1.8MHz, 900, 225, 96 kHz Prog O/P	APB	Now	1.5 k
PL130	Smartcard Interface: Compliant with the EMV Standard and ISO 7816-3	APB	Now	12.3 k
PL170	SDRAM Controller 4 Port memory controller, supporting standard SDRAM	AHB	Now	55.0 k

Note1: Gate count, NAND-2 equivalent without scan. Target Cell Library: Avant! Passport CB25 v2.1 cell library (0.25 µm)

PrimeCell Peripherals List (cont.)

Part # ARM	Description	AMBA bus Type	Availability	Gate Count
PL110	Colour LCD Controller: Colour and Mono with grey scale, Supports TFT and STN, Single and Dual Panel	AHB	Now	25.7 k
PL090	Static Memory Controller: SRAM, Flash & ROM	AHB	Now	10.5 k
PL180	Media Card Interface: Supporting MMC and SD Flash Memory Cards.	APB	Now	13.5 k
PL041	Advance Audio Codec Interface (AACI): Supporting the AC'97 Codec Interface.	APB	Now	27 k
PL190	Vector Interrupt Controller: 32 Interrupts sources and 16 Vector address	AHB	Now	20 k
PL080	Dual AHB Master, DMA controller: 8 DMA Channels w/ Scatter/Gather support	AHB	Now	~30 k
PL081	Single AHB Master, DMA controller: 2 DMA Channels w/ Scatter/Gather support	AHB	Now	~ 15 k

Note1: Gate count, NAND-2 equivalent without scan. Target Cell Library: Avant! Passport CB25 v2.1 cell library (0.25 µm)

First BlueTooth Product

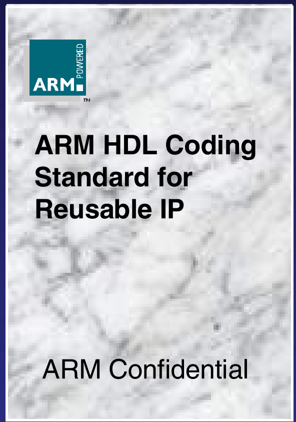
Design, Development and Delivery of Reusable IP

- *Generic Design.*
  - *It is not a point/custom solution.*
- *Designed for use in ASIC design flows.*
  - *Ensures reliable operation with Synthesis, Static Timing Analysis, ATPG/Scan Insertion, and Place & Route Tools.*
- *Flexibility*
  - *Where appropriate, flexibility has been designed into the product to enable optimisation for a specific application such as operating speed.*
  - *Delivered to ensure a good “out of box” experience.*
  - *Test benches, Design and Implementation Guides and support.*

Design for Reusable Soft IP

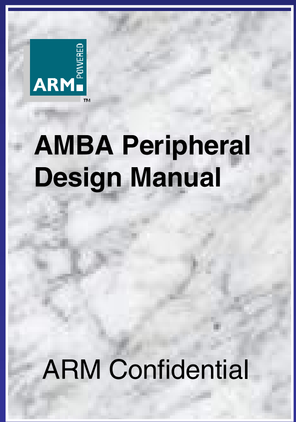
ASIC Design Rules for Reusable IP

Design rules for the creation of reusable synthesizable IP.



ARM HDL Coding Standard for Reusable IP

Rules for the development of consistent HDL for re-usable IP.

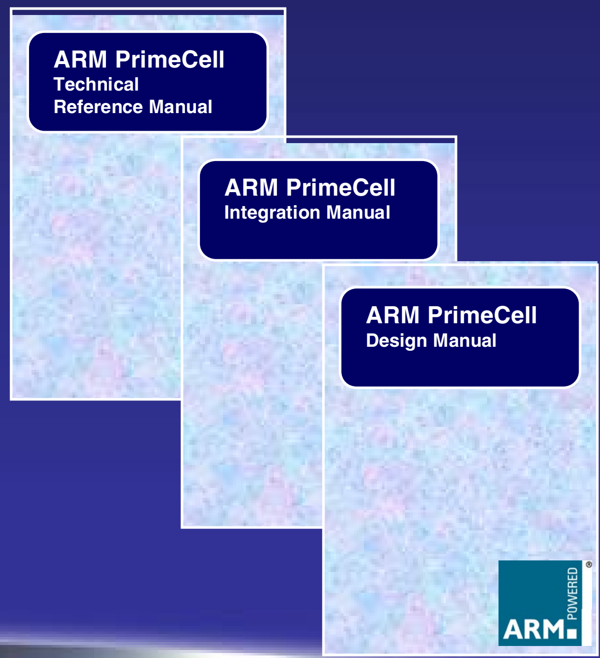


AMBA Peripheral Design Manual

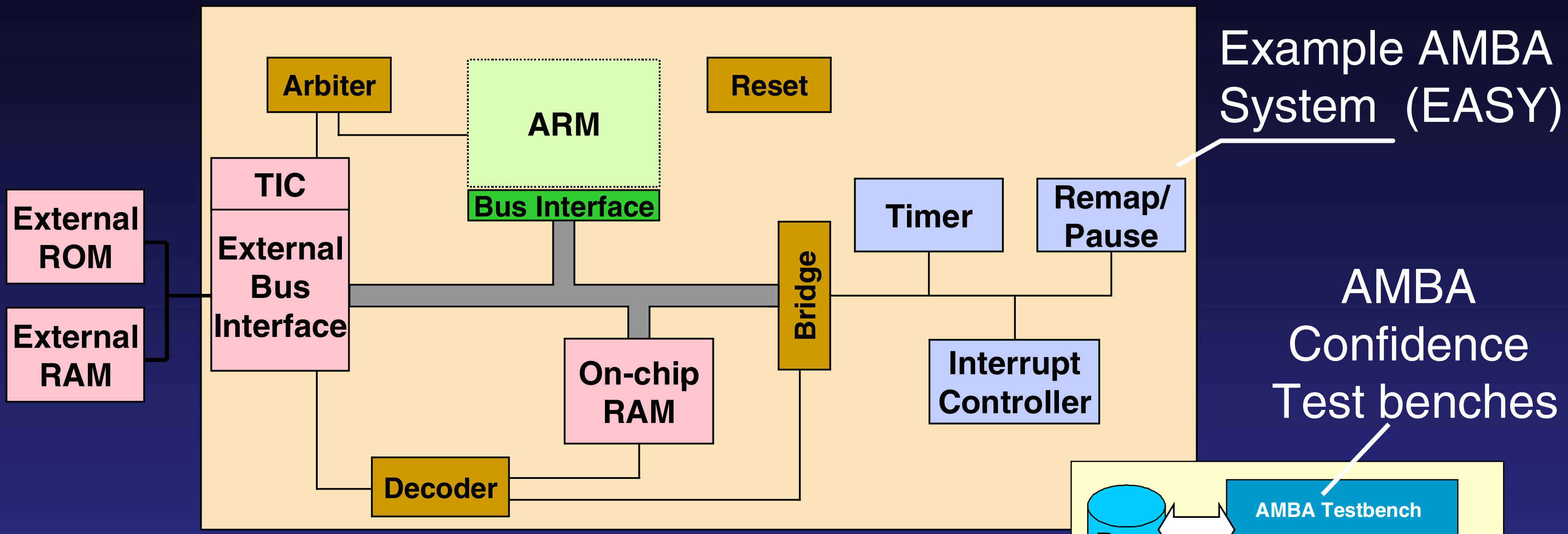
Rules to ensure functionality and consistency of ARM AMBA peripherals.

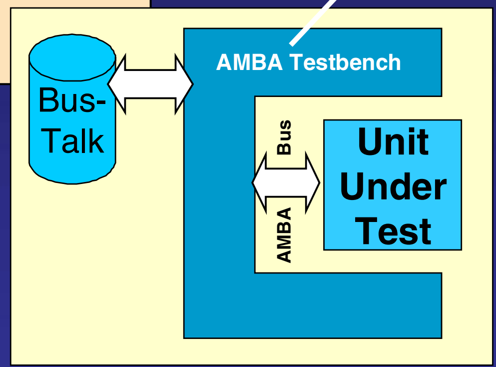
ARM PrimeCell Deliverables

- Synthesizable VHDL / Verilog code
- Synthesis scripts and timing constraints
- Functional verification test bench and source code
- Production test, source code
- Documentation:
  - Technical Reference Manual
  - Integration Manual
  - Design Manual



AMBA Design Kit



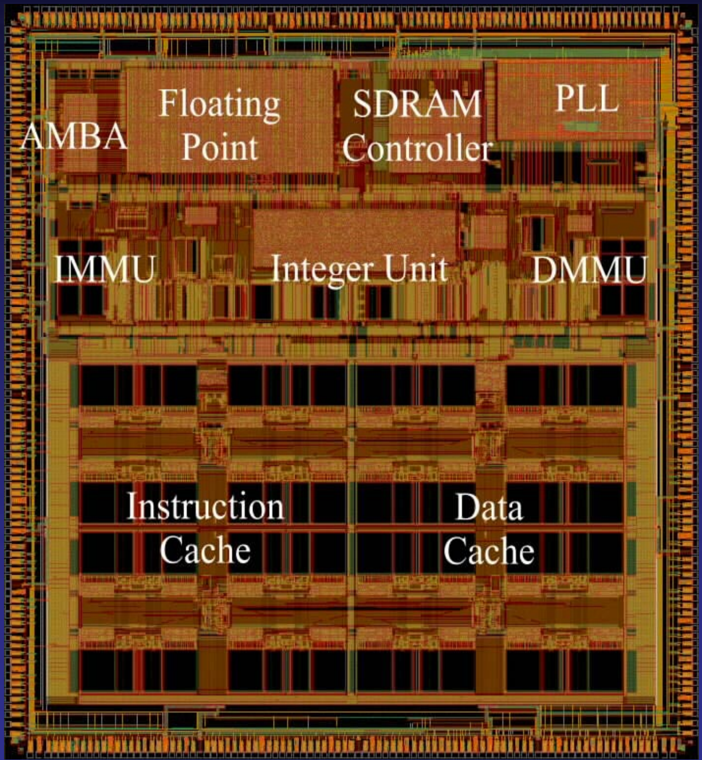


- *An Example AMBA System (EASY)*
- *Supplied in HDL (Verilog & VHDL) for easy development*
- *Reference Peripherals*
- *AMBA Compliance test-benches*
- *Enables AMBA SoC & Peripheral Development*
- *Individual macrocells can be tested in isolation at system speed*

Test Benches

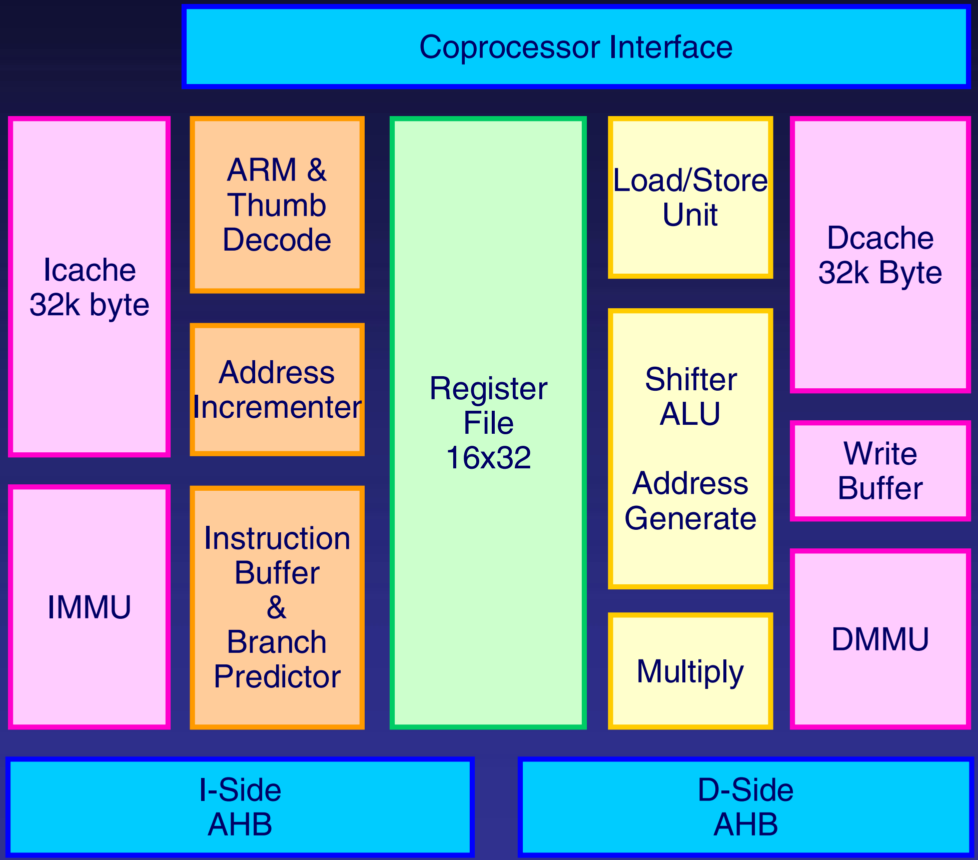
- *A simulation test bench and test validation code are supplied for each peripheral.*
- *Gives assurance that the implementation works.*
- *Reduces evaluation and test time.*
- *Operate with leading RTL Simulators.*
- *Checks for AMBA compliance.*

ARM10 Silicon



- *Test-chip, includes:*
  - *ARM1020E core*
  - *VFP10*
- *Taped out October 1999*
- *First Silicon Jan 4th 2000*

ARM1020E Core



- *ARM v5TE ISA*
  - *32-bit RISC*
  - *includes DSP extensions*
- *Architecture enhancements*
  - *6 stage pipeline*
  - *Branch Prediction*
  - *Hit-under-miss caches*
  - *Separate load-store unit*
  - *8-entry write buffer*
- *Coprocessor (VFP10)*
  - *64-bit interface & registers*
  - *IEEE floating point (single & double)*
  - *vector operations*
- *Dual 64-bit AHB bus interface*
- *Supports Embedded Trace (ETM10)*

ARM10E Architecture Enhancements

- *Branch Prediction*
  - *eliminates 70% of branches on typical code sequences*
- *Separate Load/Store Unit*
  - *64-bit path to register bank - load two registers simultaneously*
  - *Load & Store in parallel with ALU operations*
  - *highly effective LDM & STM at procedure entry and exit*
- *Hit-Under-Miss Caches*
  - *Cache continues to serve requests with up to one miss pending*
  - *significantly reduces pipe-line stalls*
- *Write Buffer*
  - *holds up to 8 double-words (16 register values)*
  - *buffer STM of complete register bank for fast interrupt entry / context switch*

High Performance Features

- *64-bit Accesses to On-chip I & D Caches*
  - *Fetch two instructions/cycle*
  - *Load/Store two registers/cycle (LDM/STM)*
- *Dual 64-bit fast AHB Bus*
  - *separate busses for Instruction & Data*
  - *>1 Gbyte/sec bandwidth @ 200MHz (each)*
  - *Split transaction extensions*
- *64-bit Coprocessor Interface*
  - *Load/Store double-precision operands in one cycle*
- *32-bit Integer datapath saves Area & Power*
  - *ARM1020E only 17.5mm² & 0.7mW/mips in 0.15um*

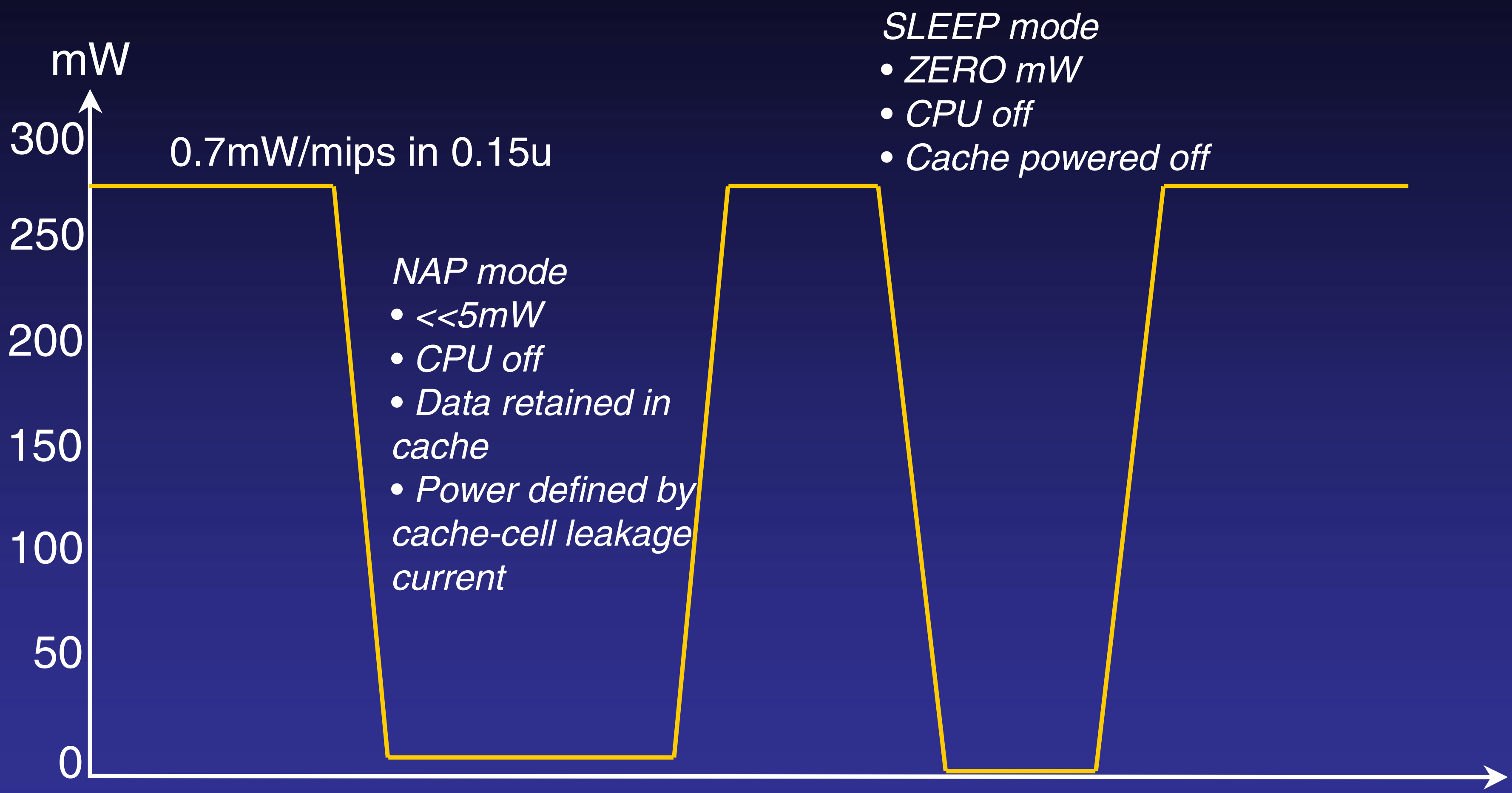
ARM1020E Core

- *Highest Performance ARM Processor Core*
- *ARM10E plus dual 32k caches & MMU*
- *50% more instructions executed per cycle than ARM920T running real code*
- *ARM DSP instruction set extensions (v5TE)*
- *Support for Vector Floating Point Coprocessor (VFP10)*
- *New Power-saving Modes - target high performance process*
- *Tolerance to data cache misses for sustained performance*
- *Support for at-speed embedded trace (ETM10)*

Delivering Power Efficient Performance

- *ARM1020E only 0.7mW/MIPS (Dhrystone)*
  - *Hard Macro = Efficient architecture, circuits, and layout*
  - *Clock gating and dynamic power reduction*
- *New power-down modes:*
  - *take advantage of high performance <0.18µm processes with high leakage currents*
  - *NAP: power-down core, but preserve state in caches typically <<5mW (dual 32k caches, depends on cache cell leakage)*
  - *SLEEP: power-down entire ARM1020E, state must be stored to memory*
    ZERO power dissipation

Power-Saving Modes



ARM10E Family : 390MIPS + DSP

VFP10

- 600Mflops Vector Floating Point coprocessor
- 1.6mm² (0.15µm)

VFP10

MMU
16K/32K Cache
ARM10EJ
ETM10 i/face
Dual AHB

Jazelle Enhanced

ARM1022E

- Platform OS core
- 300 MHz (0.15µm w/c)
- 12mm² (0.15µm)
- Hard macro IP

MMU
2x16K Cache
ARM10E
ETM10 i/face
Dual AHB

ARM102xEJ

- Platform OS core
- >400 MHz (0.13µm w/c)
- Hard macro IP

MMU
2x32K Cache
ARM10E
ETM10 i/face
Dual AHB

ARM1020E

- Platform OS core
- 300 MHz (0.15µm w/c)
- 17.5mm² (0.15µm)
- Hard macro IP

- ARM v5TE ISA
- Thumb Code Compression
- Debug support
- 6-stage pipeline
- Dual cache architecture
- Dual 64-bit AHB bus
- Optional VFP coprocessor
- 1.3 MIPS/MHz
- low power: 0.7mW/MIPS

ARM11 Preview

- Highest performance licensable ARM
  - available from multiple vendors
- 700-1000MIPS depending on
  - target process
  - aggressiveness of implementation
- Based on the new v6 architecture
  - backwards compatible with v5TE
  - substantial architectural improvements to deliver higher performance in real systems
- High performance, low cost, low power

ARM11 Preview

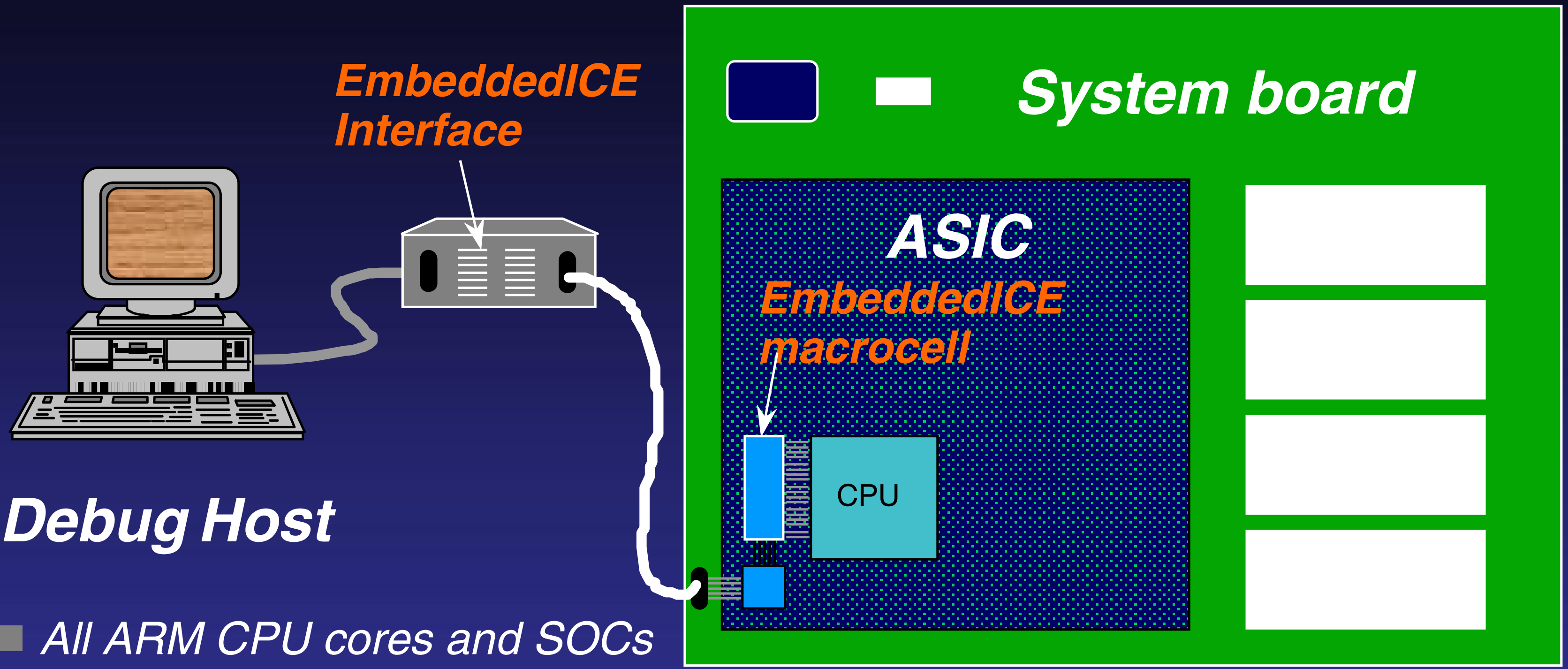
- 8-stage pipeline
- Non-Blocking Data Cache
- Virtually Indexed/Physically Addressed Caches
- MMU managed using micro TLBs
- 64-bit interfaces to both I & D Caches
- Static and dynamic branch prediction
- Microarchitectural Power improvements

ARM Architecture V6

Announced at Microprocessor Forum in October 2000

- *Media instruction set to accelerate audio and video applications*
- *SIMD arithmetic - 8-bit and 16-bit SIMD*
  - *Performs processing of multiple pixel and coefficient values in parallel*
  - *Includes instructions to pack and unpack 8-bit to 16-bit data*
  - *Faster sum-of-absolute differences for video encode*
- *Dual MAC*
  - *Two 16x16 MAC instructions per cycle to increase DSP performance*
  - *Reduced cycle times for 32x32 MAC for CD-quality audio*
- *Saturation instructions*
  - *Saturation to all common media bit boundaries (16-bit, 9-bit, 8-bit, 5-bit) for standards conformance and accelerated pixel plotting*

Embedded ICE



- *All ARM CPU cores and SOCs support the JTAG interface (IEEE 1149) for production testing and on-chip debug.*
- *Embedded ICE debug architecture provides most of the functionality of a traditional ICE without requiring any additional pins - extends functionality of the existing five-pin JTAG interface.*

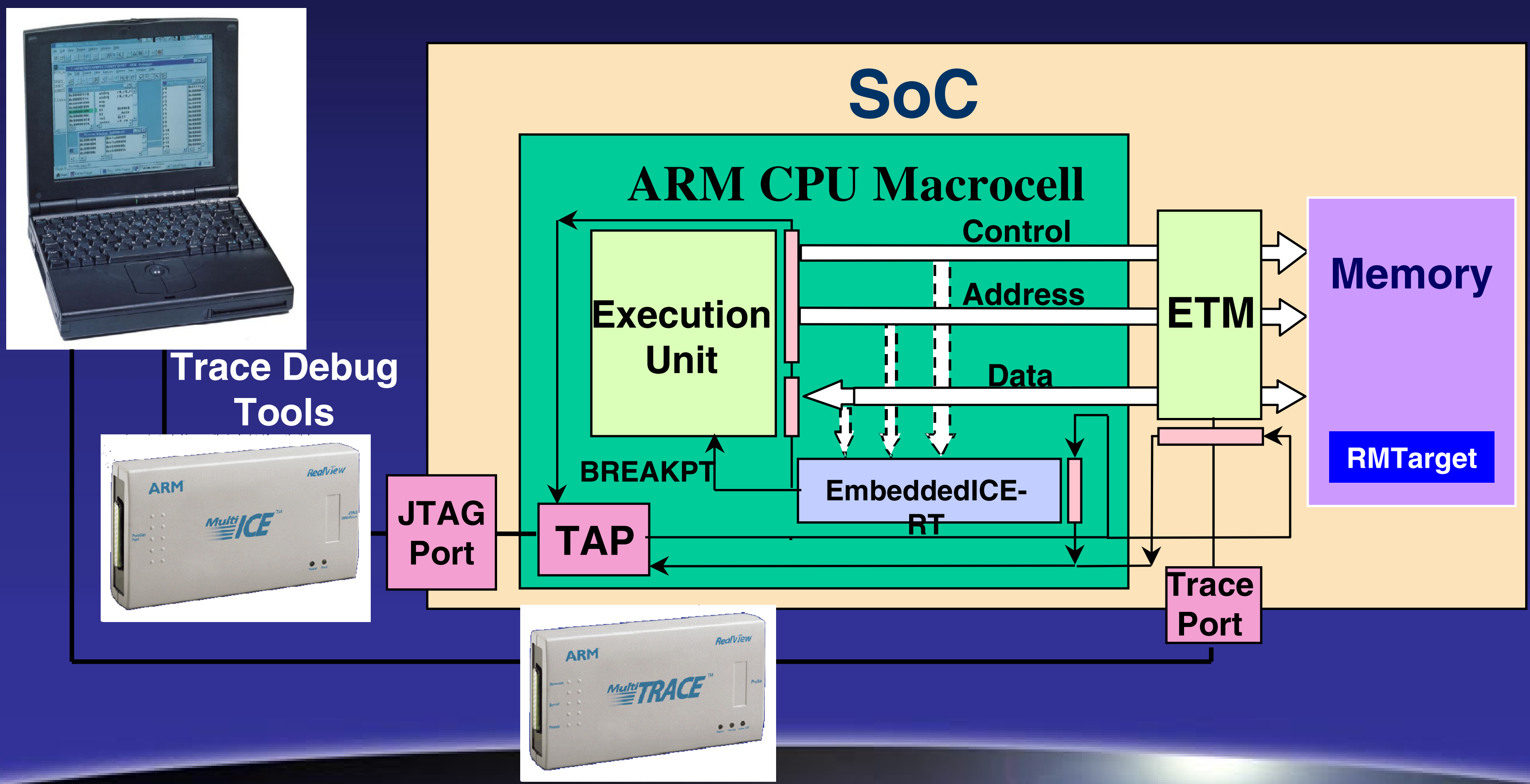
"small" amount of on-chip logic allows debug and communication via already assigned test pins.

Embedded ICE Debug Capabilities

- *Full control of CPU - inspection and modification*
- *Full control of system state - inspection and modification*
- *Uses the existing JTAG interface - no extra pins required*
- *Software debug facility providing ROM and RAM breakpoints, watchpoints, and software download.*
- *CPU stops when breakpoint or watchpoint is detected*
- *Enhanced with hardware exception catching and single stepping.*
- *ARM debugger provides source level symbolic debug of C and C++ code.*

RealTrace™

- *Run at full processor speed - real-time observation/trace of address,data and control bus activity*
- *100% non-intrusive tracing*
- *Uses the existing JTAG interface*



Real-Time Debug
On-chip Solutions

- *Run at processor speed*
- *100% non-intrusive tracing*
- *Scalable solution for multiprocessor devices*
- *Standard tools for all SOCs*
- *Easy, reliable and small interconnect to target*

With Embedded ICE and Embedded Trace, the SOC designer has all the facilities offerred by traditional ICE tools. Full visibility of the real-time behaviour of the application with the ability to set breakpoints and inspect and modify processor and system state.

Embedded Trace Macrocell
Functional Capability

Interface bandwidth is reduced using intelligent trace compression

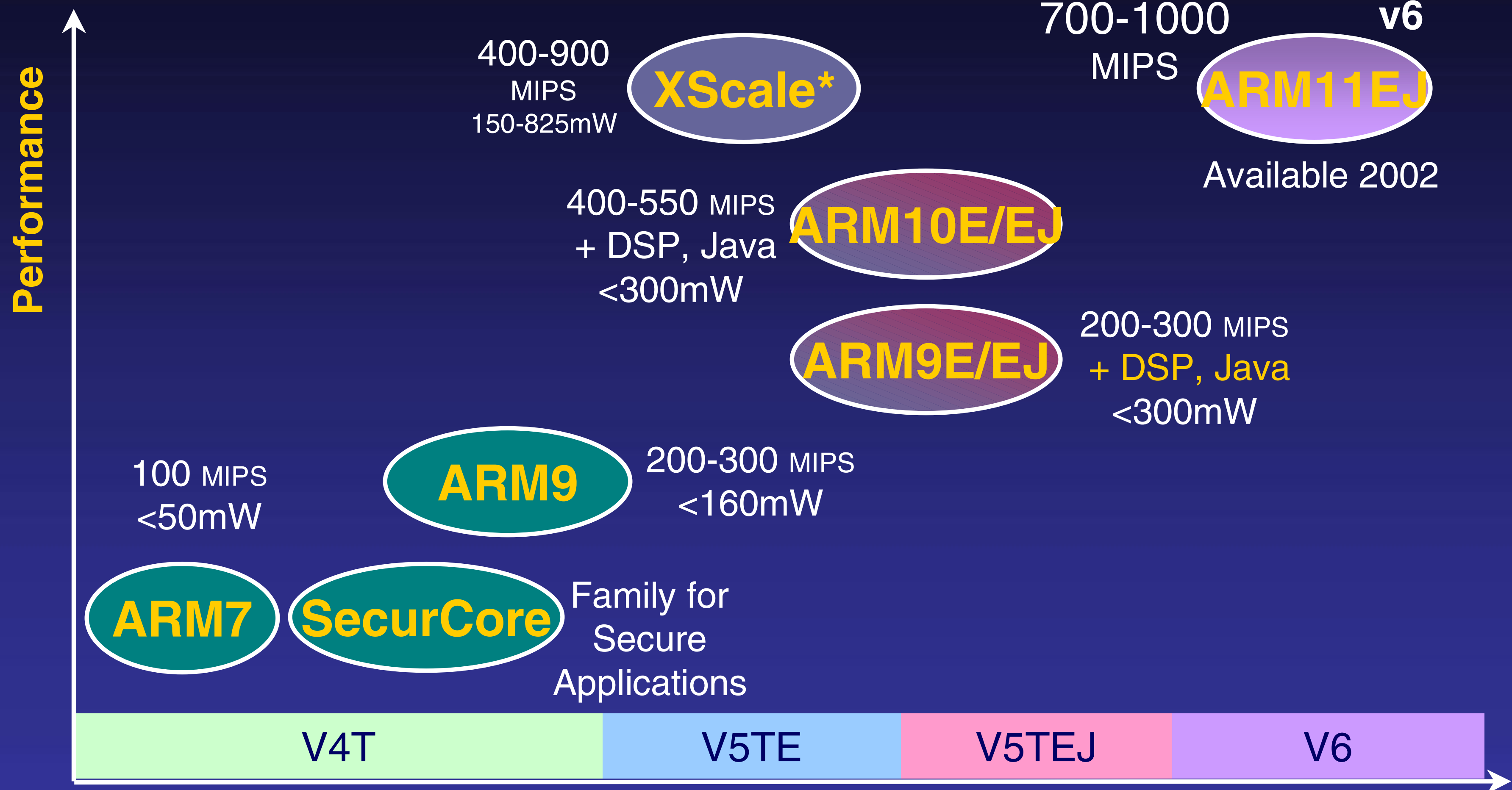
- *Capture indirect branches (Instruction trace)*
- *Capture data & address, reads or writes (Data trace)*
- *Set filter conditions to qualify which data is captured*
- *Generate external trigger from complex sequential conditions*
- *Compress data and buffer in on-chip FIFO*
- *Non-intrusively transfer data off-chip via trace port*
- *Provides cycle accurate trace*

Validation/Verification

Multiple Approach - typically 70% of current project effort

- *Deterministic self-checking testcases/regression suites*
- *Random instruction/stimulus generators*
- *Verisity Specman*
  - *Self-checking transaction-based testbenches*
  - *Protocol checking (e.g. AMBA)*
  - *Functional coverage analysis*
- *Code coverage - Verification Navigator*
- *Hardware emulation*
- *FPGA prototyping - allows for quick booting of an OS/regression runs*
- *Currently investigating formal verification - property and equivalence checking tools*

ARM Cores: 100MIPS - 1000MIPS



*XScale: Designed & Manufactured by Intel under license

Home

Future Research

- *64-bit processors*
  - *What is 64-bit? 64-bit datapath or interfaces (bandwidth)*
  - *ARM10 and ARM11 have 64-bit interfaces between the core and caches and between caches and L2 memory.*
  - *Currently few embedded applications require 64-bit ALU*
- *Superscalar processors*
  - *Currently investigating an efficient architecture for embedded applications.*
- *Asynchronous processor design*
  - *7TDMI Aysnchronous processor design completed*
- *VLIW processors*
  - *Larger architectural change, transfers control problem to software - currently investigating.*

The End

Thank you